

**Modified Enlarged 24pt**  
**OXFORD CAMBRIDGE AND RSA EXAMINATIONS**

**Friday 24 June 2022 – Morning**

**A Level Computer Science**

**H446/02 Algorithms and programming**

**Time allowed: 2 hours 30 minutes**  
**plus your additional time allowance**

**YOU CAN USE:**

**a ruler (cm/mm)**

**an HB pencil**

**DO NOT USE:**

**a calculator**

**Please write clearly in black ink.**

**Centre number**

--	--	--	--	--

**Candidate number**

--	--	--	--

**First name(s)** \_\_\_\_\_

**Last name** \_\_\_\_\_

**READ INSTRUCTIONS OVERLEAF**



## **INSTRUCTIONS**

**Use black ink. You can use an HB pencil, but only for graphs and diagrams.**

**Write your answer to each question in the space provided. If you need extra space use the lined pages at the end of this booklet. The question numbers must be clearly shown.**

**Answer ALL the questions.**

## **INFORMATION**

**The total mark for this paper is 140.**

**The marks for each question are shown in brackets [ ].**

**Quality of extended response will be assessed in questions marked with an asterisk (\*).**

## **ADVICE**

**Read each question carefully before you start your answer.**

**BLANK PAGE**

## **SECTION A**

**Answer ALL the questions.**

**1 A computer program stores data in an array named words.**

**(a) The data in the array needs to be searched for a value that the user inputs.**

**(i) One example of a searching algorithm is a binary search.**

**Identify the precondition for a binary search.**

---

---

**[1]**

**(ii) A second example of a searching algorithm is a linear search.**

**Describe how a linear search works.**

---

---

---

---

---

---

---

---

---

---

**[4]**

(b) The array words is defined as a global variable and contains these values:

"house"	"boat"	"car"	"telephone"	"garden"
---------	--------	-------	-------------	----------

"spice"	"elephant"
---------	------------

6 The pseudocode function useWords () here uses the global array words.

The number of words in the array words is passed as a parameter.

```
function useWords (numberOfWords : byVal)
```

```
    contents = ""
```

```
for count = 0 to numberOfWords - 1
    contents = contents + words[count] + " "
next count
return contents
endfunction
```

**(i) Identify TWO variables in the function `useWords()` .**

**1** \_\_\_\_\_

**2** \_\_\_\_\_

**[2]**

**(ii) `numberOfWords` is a parameter passed by value.**

**Describe the difference between passing a parameter by value and by reference.**

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ **[2]**



- (iii) Rewrite the function `useWords ()` to use a while loop instead of a for loop.

The function header and close have been written for you.

Write your answer using pseudocode or program code.

```
function  
useWords (numberOfWords :  
byVal)
```

---

---

---

---

---

---

---

---

---

```
endfunction [4]
```

**(c) Give ONE benefit and ONE drawback of declaring the array as a global variable instead of a local variable.**

**Benefit** \_\_\_\_\_

\_\_\_\_\_

**Drawback** \_\_\_\_\_

\_\_\_\_\_

**[2]**

**(d) Describe ONE feature of an Integrated Development Environment (IDE) that can be used to help write the program AND ONE feature that can be used to help test the program.**

**Write** \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**Test** \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**[4]**

**(e) Functions and procedures are reusable components.**

**Give TWO benefits of writing a program with reusable components.**

**1** \_\_\_\_\_

\_\_\_\_\_

**2** \_\_\_\_\_

\_\_\_\_\_

**[2]**

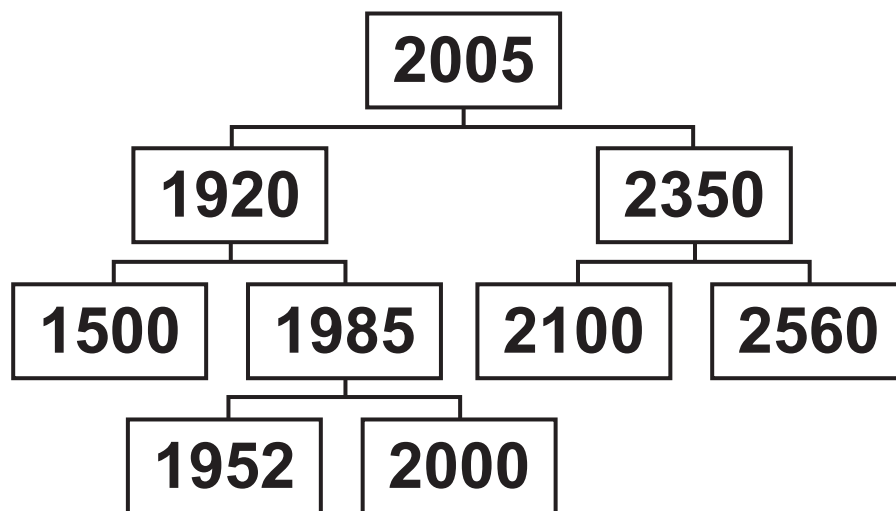
**BLANK PAGE**

- 2 A computer program is being written to store data about students.**

**FIG. 2 shows a binary search tree that stores data about students.**

**Each student is represented by their ID number. The current contents of the binary search tree are:**

**FIG. 2**



- (a) Identify the root node in the binary tree shown in FIG. 2.**

---

---

**[1]**

**(b) Identify TWO leaf nodes in the binary tree shown in FIG. 2.**

**1** \_\_\_\_\_

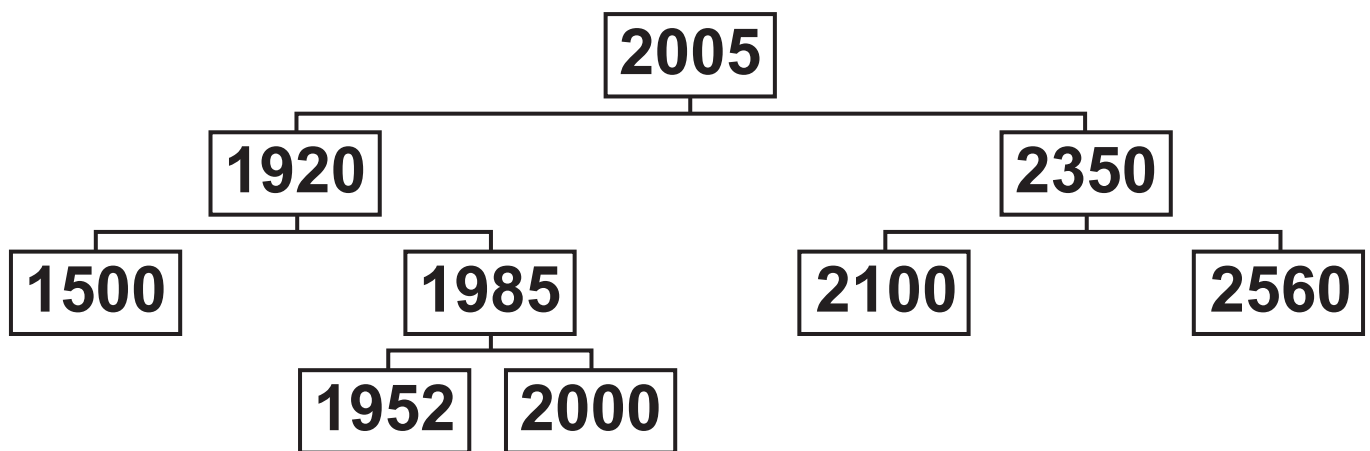
**2** \_\_\_\_\_

**[2]**

(c) Four more students are added to the binary search tree shown in FIG. 2 in this order:

1420      2050      2780      2600

Complete the binary search tree here by adding the new students to it. [4]





**(d)\*A programmer would like to traverse the binary search tree shown in FIG. 2.**

**Compare the use of a breadth-first traversal and a depth-first (post-order) traversal on the binary search tree.**

**You should include the following in your answer:**

**how each traversal works  
the order of the return values for each traversal. [9]**

---

---

---

---

---

---

---

[illegible]

[illegible]

**BLANK PAGE**

**3 A bubble sort will sort an array of 50 integer values called `numberArray`.**

**(a) State why the integer values are stored in an array instead of separate variables.**

---

---

**[1]**

(b) This bubble sort algorithm is written to sort `numberArray` into ascending numerical order.

Complete this bubble sort algorithm. [5]

```
arrayLength = _____
```

```
tempValue = 0
```

```
do
```

```
    flag = false
```

```
    for y = 0 to arrayLength -
```

```
        _____
```

```
            if numberArray[y] > numberArray[y + 1]
                then
```

```
_____ = numberArray[y]

numberArray[_____] =
numberArray[y + 1]

numberArray[y + 1] =
_____

flag = true

endif

next y

until flag == false
```

**(c) One section of `numberArray` is shown here.**

2	12	1	9	3	5	15	7
---	----	---	---	---	---	----	---

**A second sorting algorithm that could be used to sort this data is a merge sort.**

**Show how a merge sort will sort this section of the array `numberArray` into ascending numerical order. [4]**

---

---

---

---

---

---

---

---

---



---

---

---

**(d)\*Another sorting algorithm is insertion sort.**

**The number of values stored in the array `numberArray` has been reduced to 10.**

**Compare the use of bubble, merge and insertion sorts on the array `numberArray`.**

**You should include the following in your answer:**

**how each algorithm works  
the Big O complexities for each algorithm  
the suitability of each algorithm for sorting the 10 values. [12]**

---

---

---

---

---

[illegible]

[illegible]

[illegible]

**4 A programmer is developing an aeroplane simulator. The user will sit in a cockpit and the simulated environment will be displayed on screens around them.**

**(a) The programmer uses computational methods to design a solution for the program.**

**(i) Complete the table by writing a definition for each computational method. [2]**

**30**

<b>Computational Method</b>	<b>Definition</b>
<b>Abstraction</b>	
<b>Decomposition</b>	

**(ii) Give THREE potential differences between the abstracted aeroplane simulator and reality.**

**1** \_\_\_\_\_

\_\_\_\_\_

**2** \_\_\_\_\_

\_\_\_\_\_

**3** \_\_\_\_\_

\_\_\_\_\_

**[3]**

**(iii) Identify TWO reasons why abstraction is used when designing a solution to the problem.**

**1** \_\_\_\_\_

\_\_\_\_\_

**2** \_\_\_\_\_

\_\_\_\_\_

**[2]**

**(b) Describe how caching can be used in the aeroplane simulator.**

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

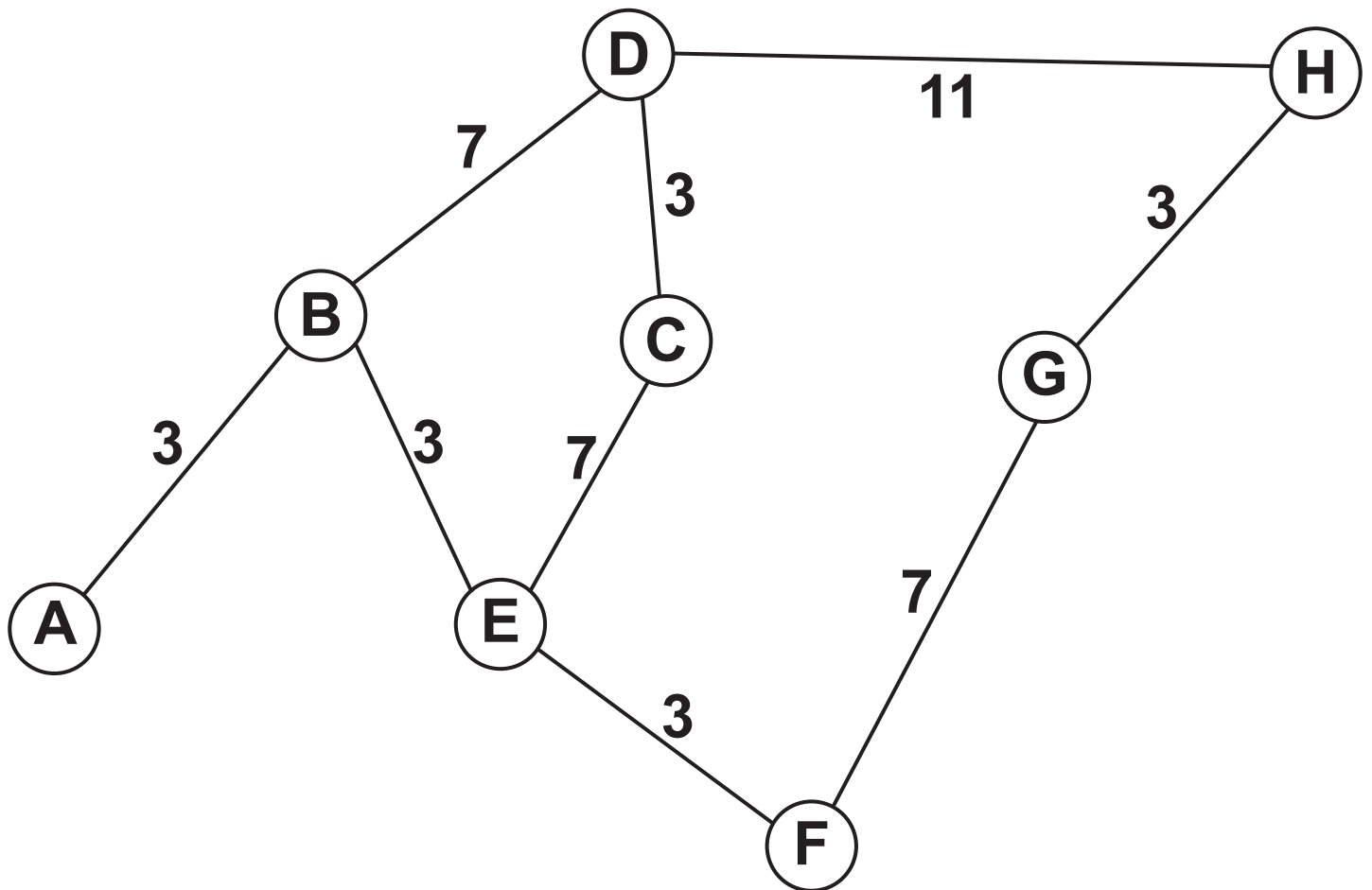
**[2]**



**BLANK PAGE**

- 5 FIG. 5 shows a graph data structure representing a small section of a parcel delivery network. Each node represents an address where deliveries need to be made. The edges show the possible routes and distances between these deliveries.

FIG. 5



- (a) (i) State ONE way a directed graph is different to an undirected graph.**

---

---

**[1]**

- (ii) State ONE way a graph data structure is different to a tree data structure.**

---

---

**[1]**

- (b) Give ONE reason why the graph is a visualisation of the problem.**

---

---

**[1]**

- (c) (i) Show how Dijkstra's algorithm can be used on the graph shown in FIG. 5 to find the shortest path from the start node A and the end node H.**

**You should state the nodes on the final path and the overall distance. Show your working.**

**You may choose to use the table below to give your answer. [6]**

---

---

---

---

---

---

---

---

---

---

---



---



---

Node	Distance travelled	Previous node

**Final path:** \_\_\_\_\_

**Distance:** \_\_\_\_\_

**(ii) Give a similarity and difference between the performance of Dijkstra's algorithm and the performance of A\* algorithm.**

**Similarity** \_\_\_\_\_

\_\_\_\_\_

**Difference** \_\_\_\_\_

\_\_\_\_\_

**[2]**

**(d) (i) State why performance modelling is used to test a system.**

---

---

**[1]**

**(ii) Describe how performance modelling can be used in the delivery system.**

---

---

---

---

**[2]**

**6 A card game uses a set of 52 standard playing cards. There are four suits; hearts, diamonds, clubs and spades. Each suit has a card with a number from; 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.**

**The card game randomly gives 2 players 7 cards each. The unallocated cards become known as the deck.**

**The players then take it in turns to turn over a card. A valid move is a card of the same suit or the same number as the last card played.**

**The winner is the first player to play all of their cards.**

**(a) One component of the game is checking if a move is valid.**

**Identify THREE other components of the game.**

**1** \_\_\_\_\_

**2** \_\_\_\_\_

**3** \_\_\_\_\_

**[3]**



**(b) A function, `checkValid()`, takes the card the player has selected, and the last card played as parameters.**

**It returns `true` if the player's move is valid and returns `false` if the player's move is not valid.**

**(i) State the reason why `checkValid()` is a function and not a procedure.**

---

---

**[1]**

- (ii) The programmer will use a branching (selection) construct to make decisions.

**Describe the decisions that will be made in the `checkValid()` function and how these change the return values.**

---

---

---

---

---

---

---

**[3]**

- (c) The cards are held in the 2D array `cards`. The first index stores the card number and the second index stores the suit, both as strings.

Write a pseudocode statement or program code to declare the array `cards`.

---

---

---

---

[2]

**7 A program uses the recursive function `calculate()`. The function is written in pseudocode.**

```
1. function calculate(number : byVal)
2.     if number == 1 then
3.         return number
4.     else
5.         return number + calculate (number - 1)
6.     endif
7. endfunction
```

- (a) (i) Give the line number in the algorithm `calculate()` where a recursive call is made.

\_\_\_\_\_ [1]

- (ii) State TWO features of any recursive algorithm.

Feature 1 \_\_\_\_\_

\_\_\_\_\_

Feature 2 \_\_\_\_\_

\_\_\_\_\_

[2]

**(b) Trace the recursive function `calculate()` and give the final return value, when the following function call is run:**

**`calculate(5)`**

**You may choose to use the table below to give your answer. [5]**

---

---

---

---

---

---

---

---

---

---

Function call	number	return
calculate (5)		

(c) Give the pseudocode function call that would return 55 from the recursive function `calculate()`.

\_\_\_\_\_

\_\_\_\_\_ [1]

**BLANK PAGE**



SECTION B

Answer ALL the questions.

8 A computer uses a stack data structure, implemented using an array, to store numbers entered by the user.

The array is zero based and has 100 locations.

(a) FIG. 8 shows the current contents of the stack and the first 9 locations of the array.

FIG. 8

pointerValue	5
--------------	---

Index	Data
8	
7	
6	
5	
4	1
3	23
2	6
1	5
0	10

- (i) The function `pop ()` removes an item from the stack.

The function `push ()` adds an item to the stack that is passed in as a parameter.

Show the contents of the stack and pointer from FIG. 8 after the following subroutines calls have run. [2]

`pop ()`

`pop ()`

`push (3)`

`push (6)`

`push (7)`

<code>pointerValue</code>	
---------------------------	--

**Index   Data**

8	
7	
6	
5	
4	
3	
2	
1	
0	

**(ii) State the purpose of  
pointerValue.**

---

---

**[1]**

**BLANK PAGE**

**(b) The stack is programmed as an object using object-oriented programming. The design for the class, its attributes and methods are shown:**

<b>class: stack</b>
<b>attributes:</b>
<b>private stackArray : Array of integer</b>
<b>private pointerValue : integer</b>
<b>methods:</b>
<b>new ()</b>
<b>function pop ()</b>
<b>function push (value)</b>

- (i) The method `pop()` returns the next value in the stack, or `-1` if the stack is empty.

Complete the pseudocode method `pop()`. [5]

```
public function pop()
```

```
    if pointerValue == _____
```

```
    then
```

```
        return _____
```

```
    else
```

```
        pointerValue =
```

```
        pointerValue _____
```

```
returnValue =  
stackArray[_____] ]  
  
return _____  
  
endif  
  
endfunction
```

- (ii) The method `push()` accepts an integer as a parameter and adds it to the top of the stack unless the stack is already full.

If the push is successful the method returns `true`.

If the push is unsuccessful due to the stack being full the method returns `false`.

Write the method `push()` using either pseudocode or program code. [6]

---

---

---

---

---

---

---

---



---

---

---

---

---

---

---

---

- (iii) The main program initialises a new object of type stack with the identifier `mathsStack`.**

**Write pseudocode or program code to declare the object.**

---

---

---

**[2]**

(iv) The main program needs to:

take numbers as input from the user  
push them onto the stack `mathsStack` until the stack  
is full  
output an appropriate message if the stack is full.

Complete the pseudocode algorithm to meet these  
requirements. [4]

```
returnValue = true
```

```
while returnValue ==
```

---

```
    returnValue = mathsStack.
```

```
    _____ (input ("Enter  
    Number" ) )
```

```
if returnValue ==  
    _____ then  
        _____ ("Stack  
full")  
    endif  
endwhile
```

**(v) The main program also needs to:**

**remove one item from the stack at a time and add this to a total  
output the total every time an item is removed  
stop removing items when either the stack is empty, or 20 items have been removed.**

**Write pseudocode or program code to meet these requirements. [8]**

---

---

---

---

---

---

---

---

---

[illegible]

**(c) The program is amended to include the use of several queue data structures.**

**(i) Describe how an array can be used to implement a queue data structure.**

---

---

---

---

---

---

---

**[3]**

**(ii)\* Discuss the use of object-oriented programming and procedural programming to create and manipulate the queue data structures.**

**You should include the following in your answer:**

**the features of object-oriented programming  
the features of procedural programming  
the benefits of using object-oriented instead of procedural programming  
when creating several queue structures. [9]**

---

---

---

---

---

[illegible]



[illegible]

---

---

---

---

---

---

---

---

**END OF QUESTION PAPER**

**ADDITIONAL ANSWER SPACE**

**If additional space is required, you should use the following lined page(s). The question number(s) must be clearly shown in the margin(s).**




[illegible]

[illegible]













## **Copyright Information**

**OCR is committed to seeking permission to reproduce all third-party content that it uses in its assessment materials. OCR has attempted to identify and contact all copyright holders whose work is used in this paper. To avoid the issue of disclosure of answer-related information to candidates, all copyright acknowledgements are reproduced in the OCR Copyright Acknowledgements Booklet. This is produced for each series of examinations and is freely available to download from our public website ([www.ocr.org.uk](http://www.ocr.org.uk)) after the live examination series.**

**If OCR has unwittingly failed to correctly acknowledge or clear any third-party content in this assessment material, OCR will be happy to correct its mistake at the earliest possible opportunity.**

**For queries or further information please contact The OCR Copyright Team, The Triangle Building, Shaftesbury Road, Cambridge CB2 8EA.**

**OCR is part of Cambridge University Press & Assessment, which is itself a department of the University of Cambridge.**